# Vers un apprentissage subquadratique pour les mélanges d'arbres

François Schnitzler, Philippe Leray, Louis Wehenkel

# Vers un apprentissage subquadratique pour les mélanges d'arbres

**François Schnitzler*** — **Philippe Leray**** — **Louis Wehenkel***

* *Département d'électricité et d'informatique & GIGA-Research*
*Université de Liège*
*Grande Traverse, 10 - B-4000 Liège - Belgique*
*{fschnitzler,L.Wehenkel}@ulg.ac.be*

** *Knowledge and Decision Team,*
*Laboratoire d'Informatique de Nantes Atlantique (LINA) UMR 6241,*
*Ecole Polytechnique de l'Université de Nantes, France*
*philippe.leray@univ-nantes.fr*

RESUME. *Dans cet article, nous comparons l'introduction d'heuristiques faibles (bootstrap, de complexité quadratique) ou plus fortes (échantillonnage aléatoire) dans l'algorithme de Chow-Liu en vue de l'apprentissage de densités de probabilité de type* mélange d'arbres de Markov. *Nos expériences empiriques sur des problèmes de grande dimension montrent que, bien que le bootstrap produise les résultats les plus précis en moyenne, d'autres heuristiques restent compétitives en terme de précision, en particulier pour des ensembles d'apprentissage de petite taille.*

ABSTRACT. *We consider randomization schemes of the Chow-Liu algorithm from weak (bagging, of quadratic complexity) to strong ones (full random sampling, of linear complexity), for learning probability density models in the form of* mixtures of Markov trees. *Our empirical study on high-dimensional synthetic problems shows that, while bagging is the most accurate scheme on average, some of the stronger randomizations remain very competitive in terms of accuracy, specially for small sample sizes.*

MOTS-CLES : *Mélanges d'arbres, Chow-Liu, apprentissage non-supervisé, perturb and combine, partitionnement approximatif*

KEYWORDS: *Mixtures of trees, Chow-Liu, unsupervised learning, perturb and combine, approximate clustering*

## 1. Contexte et motivation

Un *réseau bayésien* sur un ensemble fini $\mathcal{X} = \{X_i\}_{i=1}^n$ de $n$ variables aléatoires discrètes est un modèle probabiliste graphique composé de deux éléments (Judea, 1988). Le premier est un graphe dirigé acyclique défini sur ces variables (une bijection relie les variables et les noeuds du graphe). Ce graphe encode un ensemble de relations d'indépendance conditionnelle entre variables qui peuvent se déduire visuellement de sa structure. Le second élément est un ensemble de distributions de probabilité conditionnelle, une pour chaque variable $X_i$, conditionnellement à ses parents dans le graphe.

Un réseau bayésien encode donc une distribution de probabilité conjointe sur $\mathcal{X}$, définie par le produit de ces distributions conditionnelles, et il peut donc être exploité pour l'inférence probabiliste sur cette distribution. Cependant, l'inférence exacte est NP-difficile, tout comme l'apprentissage de la structure à partir des données si celle-ci n'est pas contrainte (Cooper, 1990).

Les *arbres de Markov* constituent un intéressant sous-ensemble des réseaux bayésiens. Ils possèdent en effet un squelette (c-à-d la structure non-dirigée du graphe) acyclique et chaque noeud du graphe possède au maximum un seul parent (Judea, 1988). La complexité algorithmique des opérations d'inférence est linéaire en le nombre de variables pour un arbre, et l'apprentissage de la structure optimale (au sens du maximum de vraisemblance sur les données) par l'algorithme de *Chow-Liu* essentiellement quadratique en $n$ (Chow *et al.*, 1968). Les contraintes sur la structure des arbres de Markov limitent cependant leur capacité de modélisation, et, en pratique, ils sont souvent inadéquats pour représenter une distribution.

Un mélange de modèles de taille $m$ consiste en un ensemble de $m$ différents modèles probabilistes, chacun défini sur la totalité de l'ensemble $\mathcal{X}$ et associé à un poids positif $w_i$ sous la contrainte $\sum_{i=1}^m w_i = 1$.

Un *Mélange d'Arbres* (Meila *et al.*, 2001) est un mélange où chaque terme $T_i$ du modèle est un arbre de Markov défini sur $\mathcal{X}$. La probabilité $P_{MA}(\mathbf{x})$ d'un événement $\mathbf{x}$ selon un modèle de mélange d'arbres est égale à la somme pondérée des probabilités de cet événement selon chaque terme individuel du mélange ($P_{T_i}(\mathbf{x})$) :

$$P_{MA}(\mathbf{x}) = \sum_{i=1}^{m} w_i \; P_{T_i}(\mathbf{x}). \qquad [1]$$

Un mélange d'arbres peut représenter une bien plus large classe de densités de probabilité qu'un seul arbre tout en préservant leurs intéressantes caractéristiques algorithmiques (Meila *et al.*, 2001), ce qui rend ces modèles très attractifs pour l'extension des modèles probabilistes graphiques aux problèmes de très grande dimension.

La plupart des algorithmes pour l'apprentissage d'un mélange d'arbres utilisent dans leur boucles internes l'algorithme de Chow-Liu (Meila *et al.*, 2001; Ammar *et al.*, 2009). Celui-ci étant quadratique en le nombre de variables, ces méthodes ne

passent pas encore bien à l'échelle pour les problèmes comportant des milliers voire des millions de variables, alors que des problèmes de cette taille se rencontrent de plus en plus en pratique. Dans cet article nous nous penchons donc sur les mélanges d'arbres appris à partir de versions heuristiques de l'algorithme de Chow-Liu, dans le but de réduire sa complexité algorithmique et d'améliorer sa précision pour des ensembles d'apprentissage de petite taille (c'est-à-dire réalistes).

La suite de cet article est organisée comme suit. Dans la section 2, nous décrivons les algorithmes que nous allons comparer, en particulier les versions heuristiques inspirées par l'algorithme de Chow-Liu. Dans la section 3, nous rapportons les résultats empiriques produits par ces algorithmes. Enfin, dans la section 4, nous résumons et envisageons différentes pistes pour la poursuite de ce travail.

## 2. Algorithmes d'apprentissage de structures d'arbre

Dans cette section sont décrits des algorithmes utilisant un ensemble d'apprentissage composé de $N$ observations conjointes de $n$ variables aléatoires, utilisés pour estimer les distributions conjointes entre chaque paire de variables et leur information mutuelle. Ces informations sont ensuite utilisées pour générer un arbre de Markov. L'algorithme de Chow-Liu est d'abord rappelé, puis trois versions heuristiques sont décrites par ordre croissant de complexité.

Une fois qu'une structure d'arbre a été obtenue, les distributions conditionnelles associées sont estimées à partir de l'ensemble d'apprentissage selon le principe du maximum de vraisemblance a posteriori avec un a priori bayésien non-informatif. C'est-à-dire, pour les variables binaires que nous utilisons ici, que la probabilité d'un événement $X_i = x_i | X_j = x_j$ se calcule à partir des fréquences observées conjointe $f_{ij}$ et marginale $f_j$ par :

$$P(X_i = x_i | X_j = x_j) = \frac{f_{ij} + 0.5}{f_j + 1}. \quad [2]$$

Cette étape requiert de l'ordre de $\mathcal{O}(nN)$ opérations (Ammar *et al.*, 2009).

Des poids uniformes sont systématiquement utilisés dans nos mélanges. En effet, les expériences réalisées dans (Ammar *et al.*, 2009) montrent que cela est préférable par rapport à une stratégie bayésienne, lorsque les termes du mélanges proviennent d'un ensemble de modèles correspondant bien aux données.

### 2.1. *Algorithme de Chow-Liu*

Cet algorithme apprend la structure d'un arbre de Markov qui maximise la vraisemblance de l'ensemble d'apprentissage (Chow *et al.*, 1968). Il se décompose en trois étapes :

1) calcul de la matrice (symétrique) des informations mutuelles empiriques ($I$) entre toutes les paires de variables, ce qui requiert $\mathcal{O}(n^2 N)$ opérations;

2) utilisation de cette matrice comme poids des arcs d'un graphe complet sur $\mathcal{X}$ à partir duquel est construit un *arbre couvrant de poids maximal* (MWST); ce qui demande un peu plus de $\mathcal{O}(n^2)$ opérations (voir par exemple (Chazelle, 2000));

3) orientation des arcs du MWST par le choix aléatoire d'un noeud-racine et la propagation des directions d'arcs loin de lui, en $\mathcal{O}(n)$ opérations.

### 2.2. *Génération aléatoire de structure d'arbre*

Cet algorithme génère une structure d'arbre totalement aléatoire, ce qui peut être fait en $\mathcal{O}(n)$ opérations (voir (Ammar *et al.*, 2009) pour une description plus précise de l'algorithme). Chaque structure d'arbre générée n'est en aucune manière reliée à l'ensemble d'apprentissage.

### 2.3. *Sélection aléatoire d'un sous-ensemble d'arcs*

Une manière simple d'améliorer la complexité de l'algorithme de Chow-Liu est de réduire le nombre de termes calculés lors de sa première étape. L'algorithme de sélection aléatoire d'un sous-ensemble d'arcs construit au lieu d'un graphe complet un graphe composé d'un nombre fixé a priori d'arcs (ou paires de variables) choisis arbitrairement. C'est ce graphe qui va être inspecté dans l'algorithme de MWST.

La complexité de cet algorithme est linéaire en le nombre d'arcs sélectionnés. La structure d'arbre résultante peut ne pas être connexe, et sa dépendance par rapport à l'ensemble de données augmente avec le nombre d'arcs.

### 2.4. *Algorithme heuristique de partitionnement de noeuds*

Dans cette section, une approche moins naïve est envisagée pour échantillonner les arcs potentiellement intéressants (c-à-d avec un poids important). Cette idée se base sur la ressemblance que présente le problème du MWST avec des questions comme la recherche du plus proche voisin ou du chemin le plus court défini sur un espace métrique, où un ensemble de $n$ points définit lui-aussi $n * (n-1)/2$ distances (ou arcs) entre ceux-ci. Dans un tel espace, ces problèmes peuvent parfois être résolu par des algorithmes subquadratiques (Indyk *et al.*, 1998) exploitant l'inégalité triangulaire satisfaite par les mesures de distances : quand un point A est proche d'un point B lui-même loin de C, alors A a de fortes chances d'être loin de C aussi.

Dans notre contexte, l'information mutuelle est utilisée pour mesurer la "proximité" entre variables, bien qu'il ne s'agisse pas d'une mesure de distance au sens mathématique. Cependant, si les variables composant les deux paires $\{A, B\}$ et $\{A, C\}$ sont proches en terme d'information mutuelle, alors $B$ et $C$ peuvent également

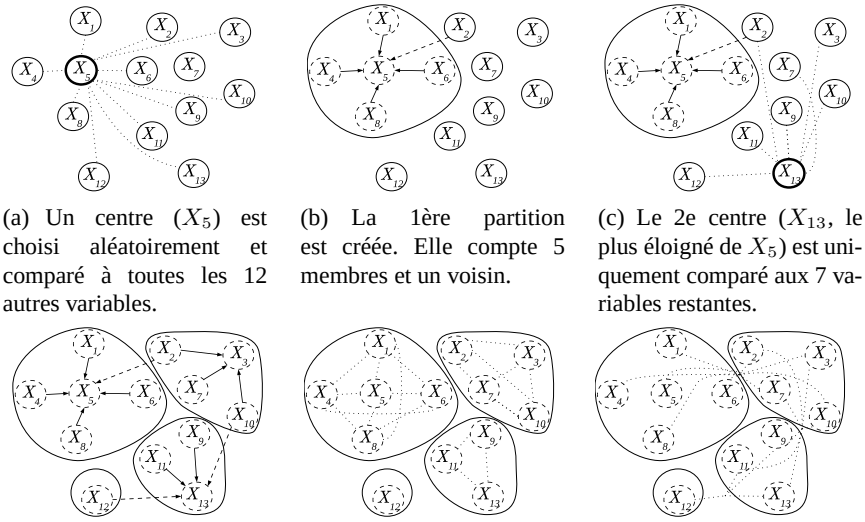
être supposées proches. Plus formellement, on peut dériver la relation : $I(B;C) \geqslant I(A;B) + I(A;C) - H(A)$, où $H(A)$ représente l'entropie de $A$.

Cet algorithme cherche à éviter le calcul de l'entièreté de la matrice $I$. Pour ce faire, il construit un partitionnement des variables en se basant sur leur information mutuelle, ce qui permet d'identifier les paires de variables qui sont proches dans cette structure et de les fournir à l'algorithme MWST de préférence à d'autres.

Ainsi qu'illustré dans la Figure 1, l'algorithme construit itérativement une succession de partitions, chacune organisée autours d'une variable centrale, jusqu'à ce que toutes les variables soient partitionnées. Le centre de la première partition est choisi aléatoirement entre toutes les variables, les suivants de manière déterministe parmi celles qui ne sont pas encore partitionnées. Chaque centre de partition (sauf le premier) est la variable candidate qui présente le moins d'information mutuelle avec tous les centres des partitions précédentes.

La construction d'une partition utilise deux seuils sur $I$ : un seuil d'appartenance à une partition ($I_P$) et un seuil de voisinage ($I_V$). L'algorithme calcule l'information mutuelle $I_i$ entre la variable choisie comme le centre de la nouvelle partition et chaque variable candidate (c-à-d qui n'est pas partionnée) $X_i$ qu'il étiquette comme :

1) un membre de la partition si $I_i > I_P$,



(a) Un centre ($X_5$) est choisi aléatoirement et comparé à toutes les 12 autres variables.

(b) La 1ère partition est créée. Elle compte 5 membres et un voisin.

(c) Le 2e centre ($X_{13}$, le plus éloigné de $X_5$) est uniquement comparé aux 7 variables restantes.

(d) Après 4 itérations, toutes les variables appartiennent à une partition.

(e) Calcul de l'information mutuelle entre variables appartenant à une partition identique.

(f) Calcul de l'information mutuelle entre variables appartenant à deux partitions voisines.

**Figure 1.** *Illustration de l'algorithme heuristique de partionnement de variables.*

2) un membre du voisinage de la partition si $I_P \geq I_i > I_V$,

3) non reliée à la partition sinon.

Fixer ces seuils peut être vu comme décider de ce qui est potentiellement indépendant et peut être exclu. La proportion de variables indépendantes exclues peut être contrôlée, vu que l'information mutuelle empirique suit dans ce cas une loi $\chi^2$ .

Dans la seconde étape de l'algorithme, l'information mutuelle de toutes les paires de variables jugées intéressantes sont évaluées, et les arcs correspondant fournis à l'algorithme MWST. Une paire est considérée intéressante si les deux variables appartiennent à des partitions identiques ou voisines. Deux partitions sont dites voisines si au moins une variable de l'une appartient au voisinage de l'autre. Outre ces paires intéressantes, tous les arcs qui ont été calculés pendant le processus de partitionnement sont également utilisés pour le calcul du MWST.

La complexité de cet algorithme se situe entre les cas linéaire et quadratique en le nombre de variables, et dépend fortement de la distribution considérée et des valeurs des seuils.

## 3. Résultats expérimentaux

Les algorithmes décrits dans la section précédente ont été appliqués à des problèmes synthétiques, d'après la méthodologie décrite dans (Ammar *et al.*, 2009). Les résultats présentés ici pour chaque $N$ sont moyennés sur 10 ensembles d'apprentissage $\times$ 10 distributions cibles définies sur 1000 variables binaires.

La méthode de la section $2.4$ a été appliquée en utilisant pour $I_P$ et $I_V$ des valeurs correspondant respectivement aux seuils 0.5% et 5% d'une distribution $\chi^2$. Pour évaluer son intérêt par rapport à la méthode de sélection aléatoire d'un sous-ensemble d'arcs de la section $2.3$, celle-ci a été contrainte d'échantillonner la même proportion (35%) d'arcs que la première. Comme point de comparaison, nous utilisons : (1) un unique arbre de Markov construit selon l'algorithme de Chow-Liu; (2) des mélanges d'arbres de structure totalement aléatoire ; (3) des mélanges construits par application de l'algorithme de Chow-Liu à des copies bootstraps de l'ensemble d'apprentissage (Ammar *et al.*, 2009). Les temps CPU (indicatifs) obtenus pour l'apprentissage de $10\times10$ mélanges de taille $m = 100$ sont donnés dans le tableau 1.

La précision des modèles appris a été évaluée en utilisant une approximation Monte-Carlo de la divergence de Kullback-Leibler (Kullback *et al.*, 1951) entre la distribution cible $P_c$ et celle apprise $P_a$, calculée par :

$$\hat{D}_{KL}(P_c||P_a) = \sum_{\mathbf{x}\sim P_c} \log \frac{P_c(\mathbf{x})}{P_a(\mathbf{x})}. \qquad [3]$$

Deux ensembles de 60 000 échantillons ont été utilisés, pour vérifier la convergence de ces estimations.

**Tableau 1.** *Temps CPU pour l'apprentissage, cumulés sur 100 ensembles de données de 1000 échantillons (MacOS X; Intel dual 2 GHz; 4GB DDR3; GCC 4.0.1).*

| arbres aléat. | échant. d'arcs aléat. | partitionnement | bootstrap |
|---|---|---|---|
| 2 063 s | 64 569 s | 59 687 s | 168 703 s |

### 3.1. *Résultats*

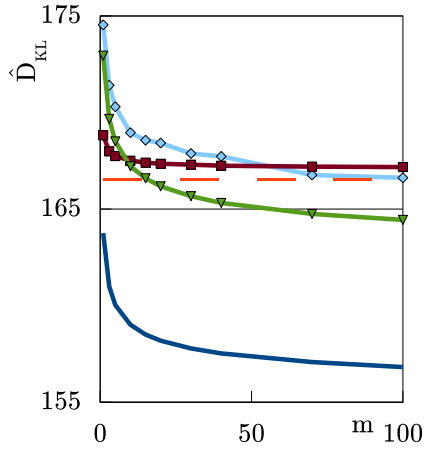
La Figure 2 présente les valeurs $\hat{D}_{KL}$ des modèles construits pour tous les algorithmes, en variant d'abord la taille $m$ des mélanges ($2(a)$), ensuite la taille $N$ de l'ensemble d'apprentissage ($2(b)$). La Figure $2(b)$ montre que les méthodes les plus sophistiquées tendent à converger plus lentement. Sur la Figure $2(b)$ on peut voir que la méthode bootstrap produit les meilleurs résultats, sauf pour $N < 50$, et que l'échantillonnage d'arcs et le partionnement des variables supplantent l'arbre de Chow-Liu pour respectivement $N < 200$ et $N < 400$.

Sur ces deux figures apparaît clairement l'intérêt de la méthode de partionnement de variables ($\bigtriangledown$ vert): la détection plus intelligente de la structure du problème conduit à une amélioration par rapport à l'échantillonnage naïf des arcs ($\Diamond$ bleu clair). Bien que l'amélioration soit faible, elle est significative, et ne dégrade pas la complexité de calcul.

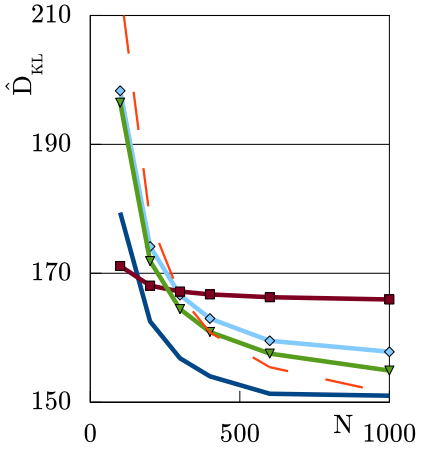
Nous suspectons que le comportement de l'algorithme de partionnement de variables dépend fortement de la structure du problème, et nous remettons l'étude de l'impact de ses paramètres à un travail futur, incorporant des données réelles. Nous pensons cependant que ses résultats seront en partie proportionnels au nombre d'arcs considérés. Pour en donner un aperçu, la Figure 3(a) présente l'effet de la modification de la proportion d'arcs considérés par la méthode de l'échantillonnage aléatoire. Sans surprise, plus le nombre d'arcs augmente, plus la courbe s'approche de celle de l'arbre de Chow-Liu. Pour un petit ensemble d'apprentissage, moins on a d'arcs, meilleur est le résultat, alors que c'est l'inverse pour des ensembles plus grands.

Les différentes méthodes présentées ici peuvent également être combinées. La Figure 3(b) présente les résultats obtenus en appliquant l'algorithme d'échantillonnage aléatoire d'arcs à des copies bootstraps de l'ensemble d'apprentissage. Les résultats sont quasiment aussi bons que pour la méthode de bootstrap pour de petits ensembles d'apprentissage. Lorsque la taille de ceux-ci augmentent, la performance se rapproche progressivement de celle de l'échantillonnage sur les données initiales, et, pour un ensemble de 1000 échantillons, est très légèrement moins bonne.

Pour vérifier la solidité de nos conclusions, la Figure 3.1 indique la variabilité de notre mesure de performance : pour chacune des 10 distributions cibles (arrangées horizontalement) et méthodes est affichée une boîte à moustaches de 20 valeurs $\hat{D}_{KL}$, obtenues à partir de 10 ensembles d'apprentissage (de taille 300) fois 2 ensembles de test (de taille 60 000). Les tendances observées précédemment se maintiennent

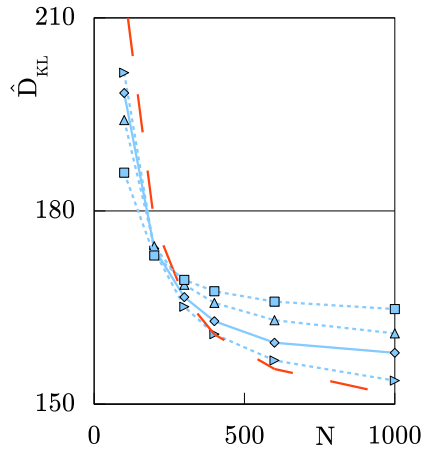



(a) Effet de la taille $m$ des mélanges ($N$ = 300).

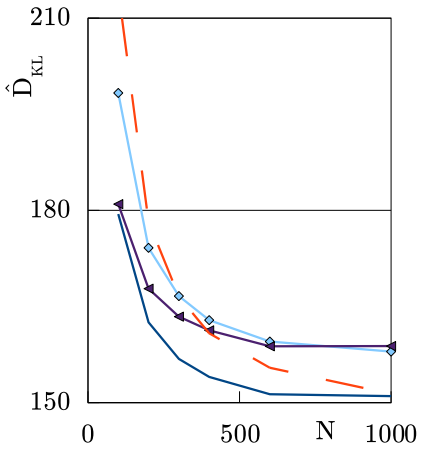



(b) Effet de la taille $N$ des ensembles d'apprentissage ($m$ = 100).

**Figure 2.** *Performance (moyenne sur 10 distributions cibles fois 10 ensembles d'apprentissage) de l'échantillonnage aléatoire d'arcs (◊ bleu clair), partionnement des variables (▽ vert), bootstrap pour les structures d'arbre (bleu foncé), structures d'arbre aléatoires (□ brun), et un unique arbre de Markov optimal (pointillé rouge). Les divergences KL sont estimées par Monte-Carlo.*




(a) Effet du nombre d'arcs considérés : 60, 35, 20, 5% (▷, ◊, △, □).




(b) Combinaison (◁ mauve) de bootstrap (bleu) et d'échantillonnage (35%) d'arcs (◊).

**Figure 3.** *Analyses plus poussées de l'algorithme d'échantillonnage aléatoire d'arcs ($m$=100). Les divergences KL sont estimées par Monte-Carlo (valeurs moyennées sur 10 distributions de 1000 variables $\times$ 10 ensembles d'apprentissage). Les pointillés rouges correspondent à un arbre de Chow-Liu.*

quasiment sans exception. Les différences sont dues à la variation des plages (sur $N$ et $m$) mettant en évidence différents comportements, et non à une modification de ceux-ci.

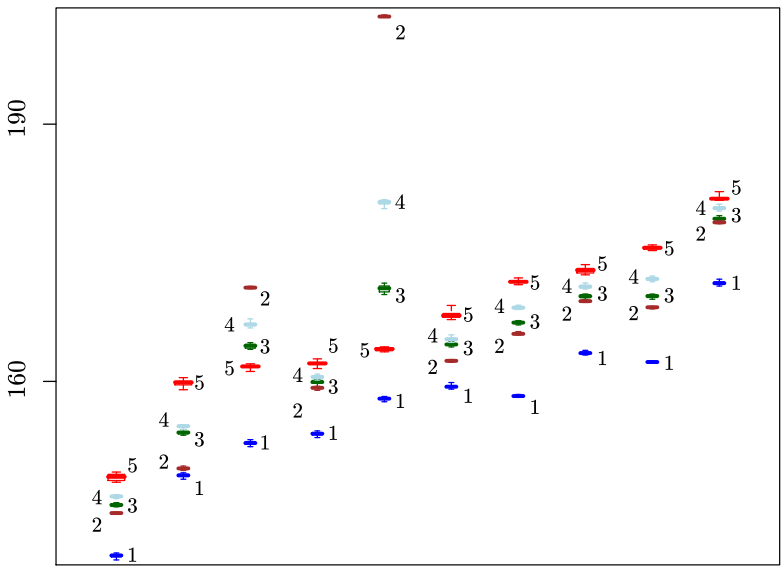



**Figure 4.** *Variation de la précision obtenue sur les 10 distributions cibles, disposées horizontalement ($m$=100, $N$=300). Chaque boîte à moustaches est définie par 20 divergences KL : 10 ensembles d'apprentissage fois 2 ensembles de test. Les algorithmes affichés sont : (1) bootstrap pour les structures d'arbre, en bleu foncé; (2) structures d'arbre aléatoires, en brun; (3) partionnement des variables, en vert; (4) échantillonnage aléatoire d'arcs, en bleu clair; (5) unique arbre de Chow-Liu, en rouge.*

## 4. Conclusion

Dans cet article nous étudions empiriquement différentes heuristiques appliquées à l'algorithme de Chow-Liu pour l'apprentissage de mélanges d'arbres. Notre étude montre que la perte en précision est liée au gain en complexité : les deux nouvelles méthodes d'échantillonnage d'arcs sont meilleures en terme de précision que l'échantillonnage de structures aléatoires, plus simple, mais pires que le bootstrap plus complexe. Nous observons également qu'une heuristique plus poussée est productive quand le nombre d'échantillons $N$ est beaucoup plus petit que le nombre de variables $n$, ce qui est la règle dans les problèmes de grandes dimensions.

Une obervation plus intéressante résulte de la comparaison de nos deux méthodes d'échantillonnage : l'exploitation de la structure du problème via un partionnement des variables conduit en effet à une amélioration significative du résultat par rapport à un échantillonnage totalement aléatoire des arcs, et ce sans détériorer la complexité.

Même si ces méthodes ne sont pas encore une alternative convaincante à l'état de l'art, des améliorations supplémentaires peuvent être envisagées. D'abord, l'espace des paramètres devrait être exploré, afin de définir une méthode pour les régler de manière adaptative. Ensuite, la sélection du ou des meilleurs modèles parmi ceux générés pourrait être une voie intéressante pour améliorer ces méthodes.

Des alternatives à la méthode de partionnement de variables peuvent également être proposées, par exemple une méthode gloutonne améliorant un arbre en comparant les variables proches dans la structure de graphe déjà construite.

Remerciements

La recherche présentée ici a pu être réalisée grâce à une bourse F.R.I.A., ainsi que le soutien de Wallonie-Bruxelles International et du Fonds de la Recherche Scientifique, du Ministère Français des Affaires étrangères et européennes, du Ministère de l'Enseignement supérieur et de la Recherche dans le cadre des Partenariats Hubert Curien. Il a également été soutenu par le réseau IUAP Biomagnet de l'office belge de la politique scientifique et le réseau d'excellence de la communauté européenne Pascal2. La responsabilité scientifique est celle des auteurs.

## 5. Bibliographie

Ammar S., Leray P., Defourny B., Wehenkel L., « Probability Density Estimation by Perturbing and Combining Tree Structured Markov Networks », *Proceedings of ECSQARU*, p. 156-167, 2009.

Chazelle B., « A minimum spanning tree algorithm with inverse-Ackermann type complexity », *J. ACM*, vol. 47, n° 6, p. 1028-1047, 2000.

Chow C., Liu C., « Approximating discrete probability distributions with dependence trees », *IEEE Trans. Inf. Theory*, vol. 14, p. 462-467, 1968.

Cooper G., « The computational complexity of probabilistic inference using bayesian belief networks », *Artificial Intelligence*, vol. 42, n° 2-3, p. 393-405, March, 1990.

Indyk P., Motwani R., « Approximate nearest neighbors: towards removing the curse of dimensionality », *STOC '98: Proceedings of the thirtieth annual ACM symposium on Theory of computing*, New York, NY, USA, p. 604-613, 1998.

Judea P., *Probabilistic reasoning in intelligent systems: networks of plausible inference*, Morgan Kaufmann, 1988.

Kullback S., Leibler R., « On Information and Sufficiency », *ANN MATH STAT*, vol. 22, n° 1, p. 79-86, 1951.

Meila M., Jordan M., « Learning with mixtures of trees », *J. Mach. Learn. Res.*, vol. 1, p. 1-48, 2001.